MINOS DAQ - An Overview

Tim Nicholls - Rutherford Appleton Lab

MINOS Online Meeting, RAL

29th January 2001

Introduction

- Aim of talk: set scene for DAQ discussions

- Topics:
  - DAQ Concepts
  - System Overview
  - Run Model (new to many)

- No detailed info / test results etc:
  - Tass will give more details in next talk
  - Final DAQ review - 20th February

DAQ Concepts

- Start with a general definition...
- “DAQ” as terminology is vague
  - Often use “readout”, “DAQ” & “online” interchangeably
  - Confusion about what belongs where abounds
- For our purposes, consider DAQ as:
  - Everything in the data chain between front-end electronics (FEE) and persistent data storage

Concepts (II)

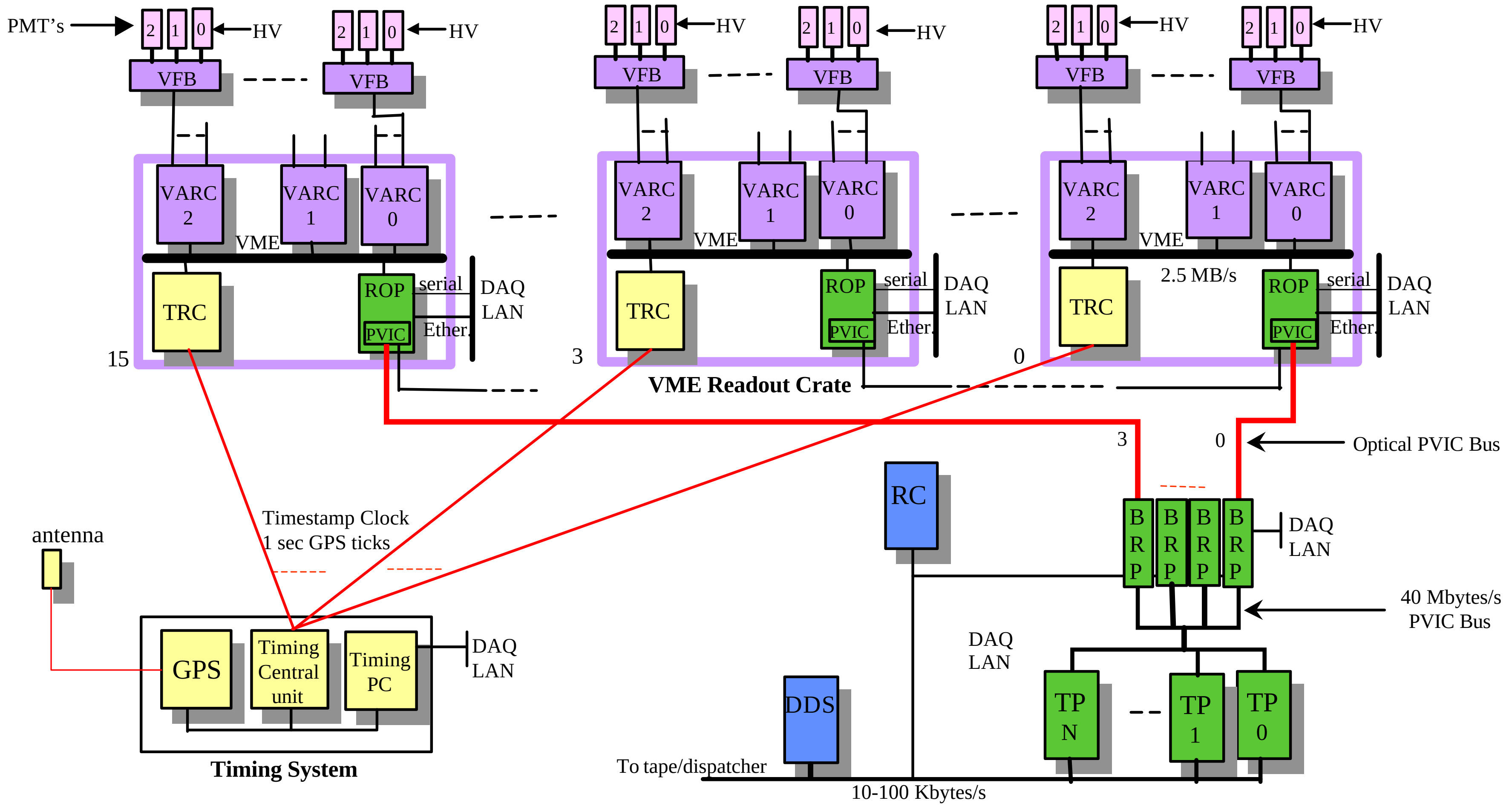
- DAQ operates continuously-live readout:
  - FEE provide alternating data buffers
  - DAQ incurs 0% 1st-order dead-time“
- Quantum” of FEE data is the Time Block:
  - TB length approx 10-50ms (tuneable)
- Build overlapping Time Frame for trigger:
  - TF length approx 0.5 - 1s (tuneable)
  - Overlap is one or more TB (trigger efficiency)

Concepts (III)

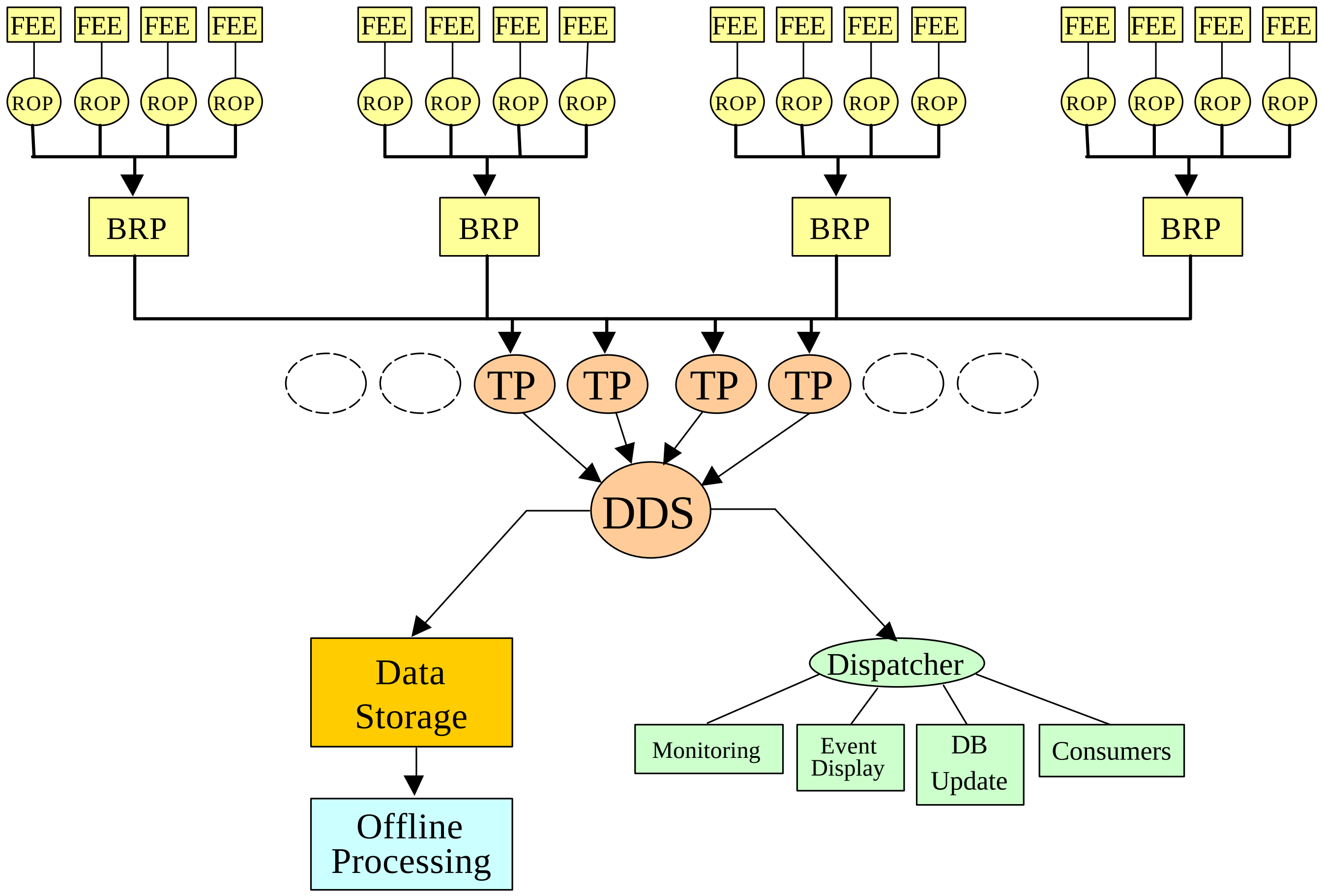
- Functional components of DAQ:
  - Readout of data from Front-end electronics Data transfer and assembly
  - Software triggering
  - Event formatting & distribution

- Same architecture at near & far detectors
  - Number of components differs
  - Software layer for FEE access changes

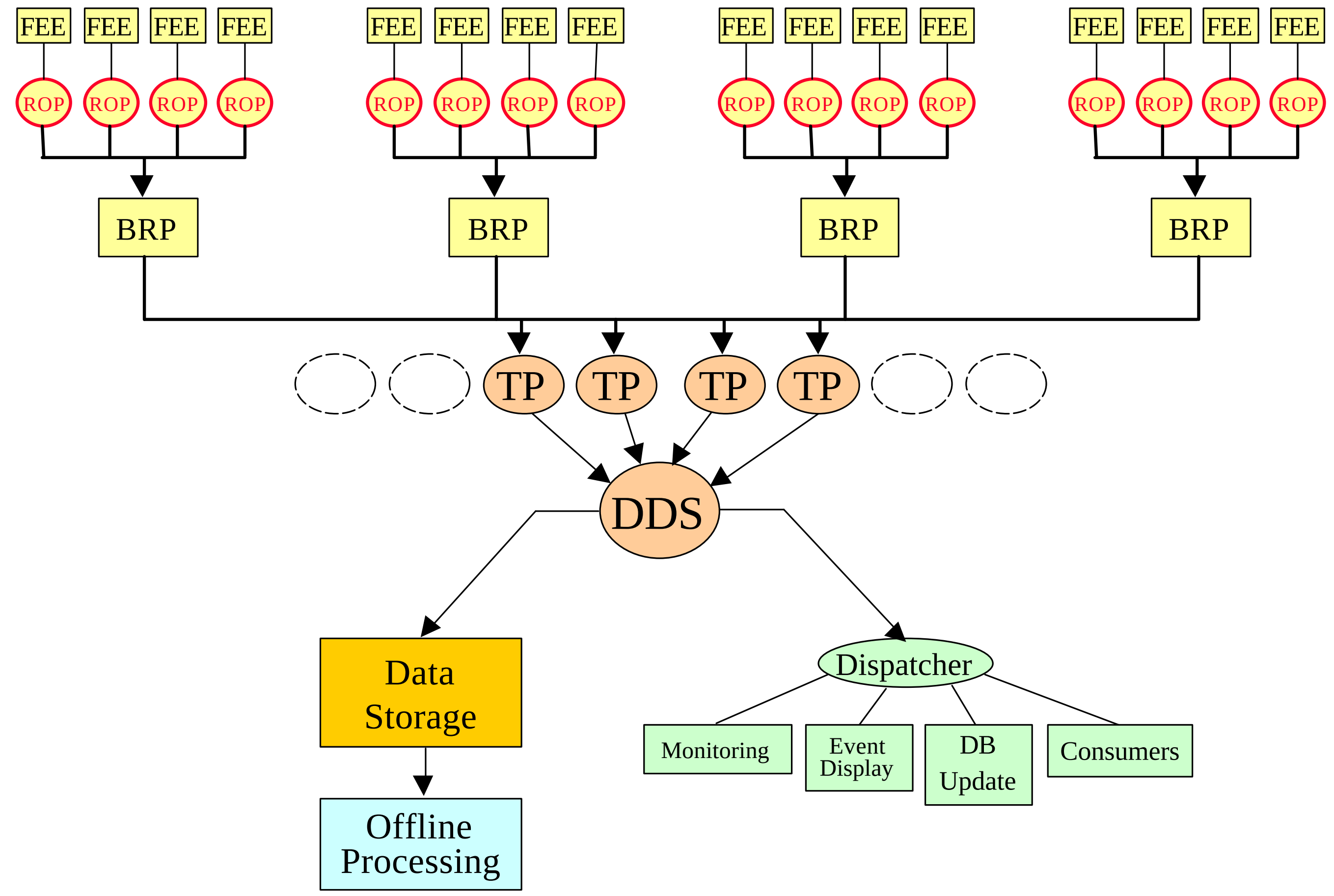
DAQ Architecture (Far)



Online Data Flow



Readout Processor

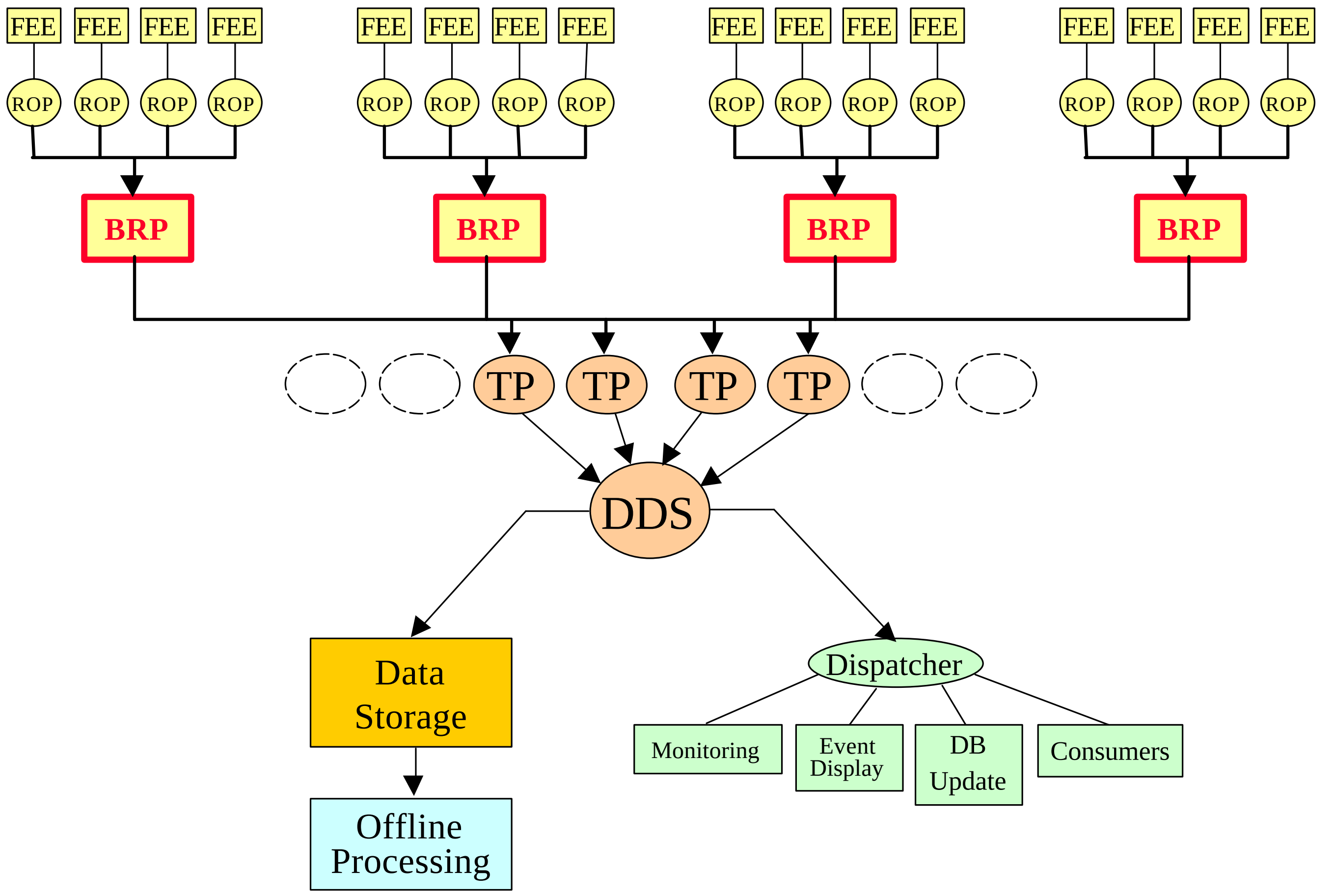


Readout Processor

- ROP provides:
  - Control & monitoring of FEE
  - Readout of data from FEE in time blocks
  - Buffering & assembly of time frames (with overlaps)
  - Transfer of built time frames over PVIC to BRPs
- 16 (8) ROPs @ far (near) detector
- 4 (2) ROPs per PVIC input branch

- Plaftorm: CES RIO3 PowerPC / VxWorks 5.4

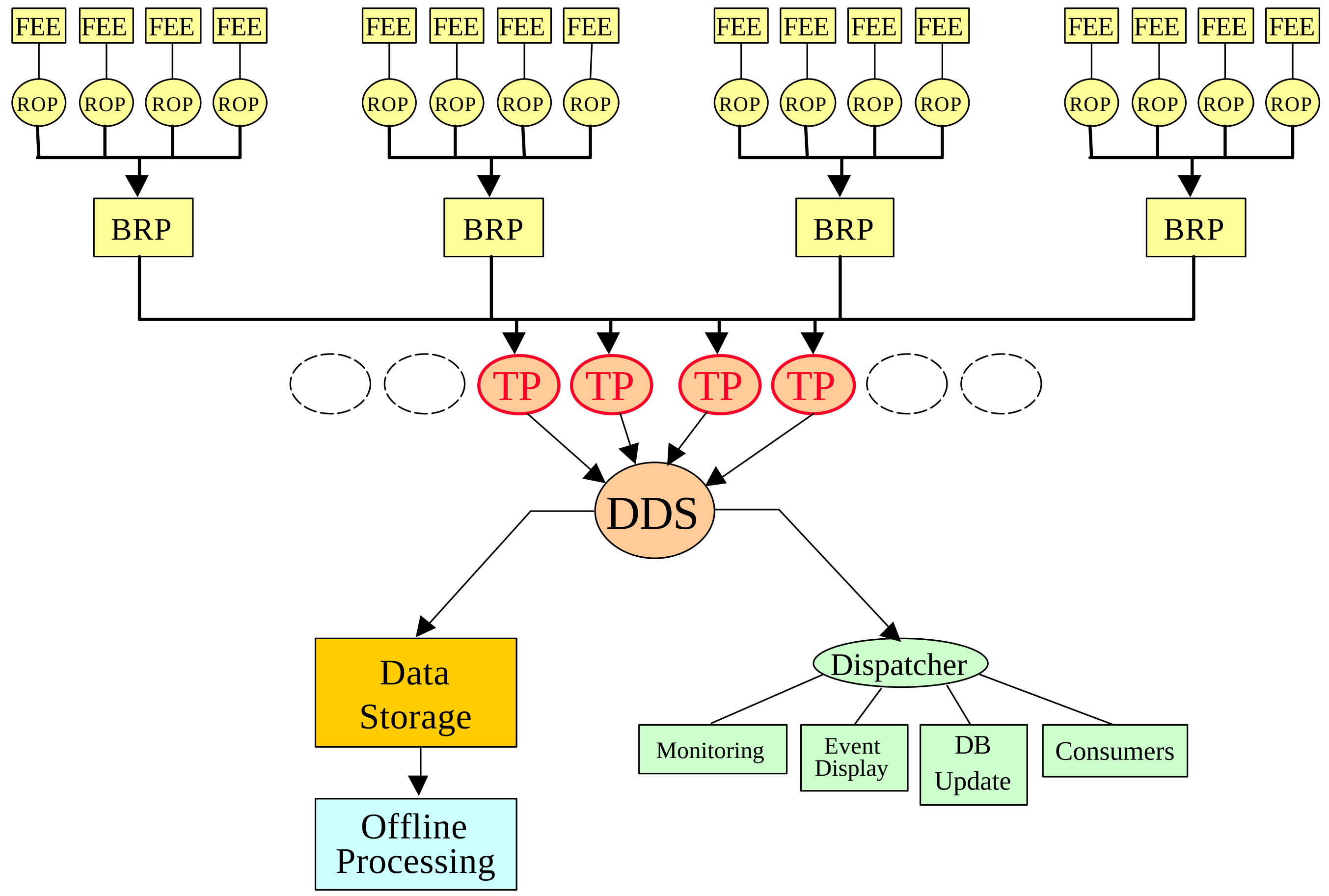
Branch Readout Processor



Branch Readout Processor

- BRP provides:
  - Communication path to ROPs/TPs via PVIC
  - Synchronisation of data transfer from ROPs and buffering of time frames
  - Transfer of time frames to TPs
  - Monitoring of data flow
- One BRP acts as “master”
  - Interface to Run Control, sequences readout
  - Can be assigned dynamically
- Platform: Linux PC

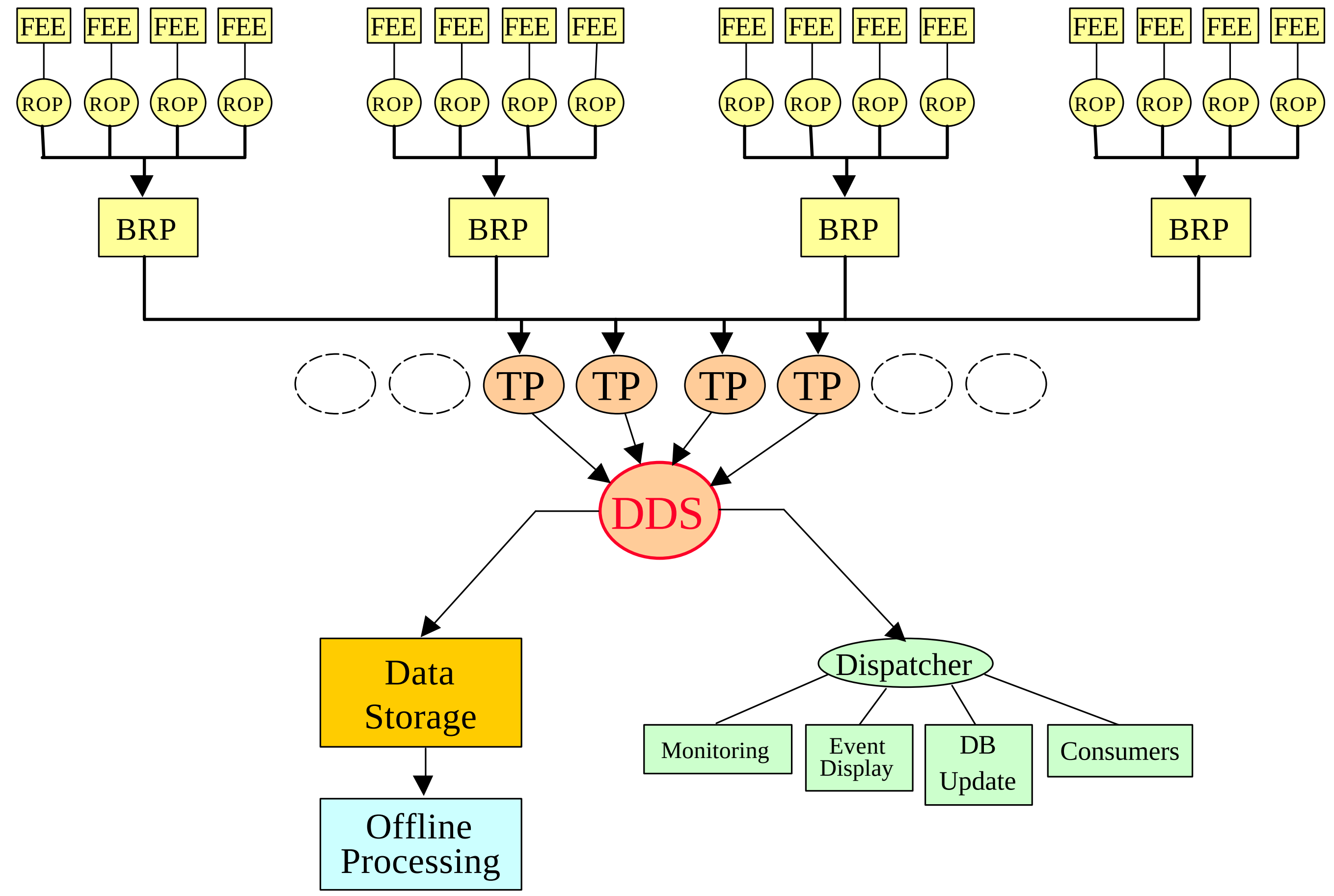
Trigger Processor



Trigger Processor

- TP provides:
  - First location where all data in TF together
  - Final time sorting of hits in time frames
  - Selection of events by trigger algorithms
  - Flasher data processing
  - Monitoring
  - Output of selected data to DDS
- TPs (max 11) daisy chained to BRPs on PVIC output branch
- Platform: Linux PC

Data Distribution System



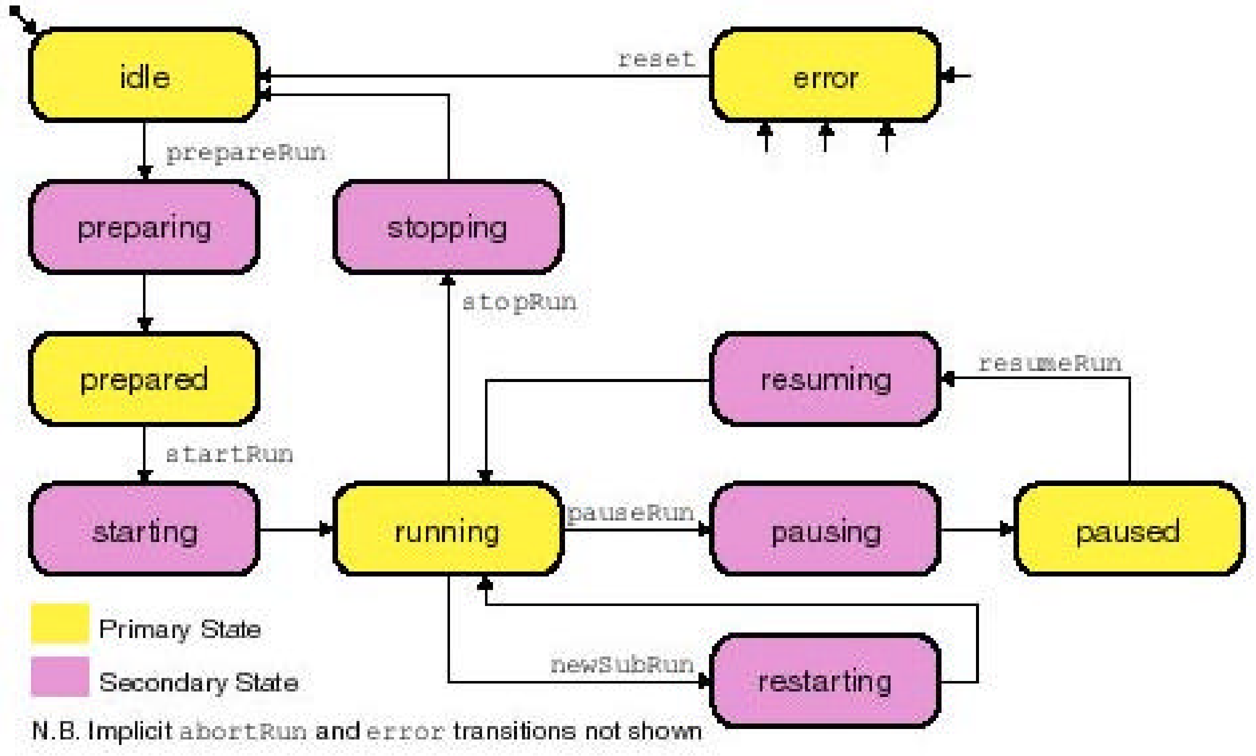
Data Distribution System

- DDS provides:
  - Reception of data from TPs via DAQ LAN (rate low)
  - Removal of duplicate triggers caused by overlaps
  - Formatting of raw data into ROOT files
  - Shipping of data for permanent storage
  - File access for dispatcher
- Platform: Linux PC

DAQ Run Model

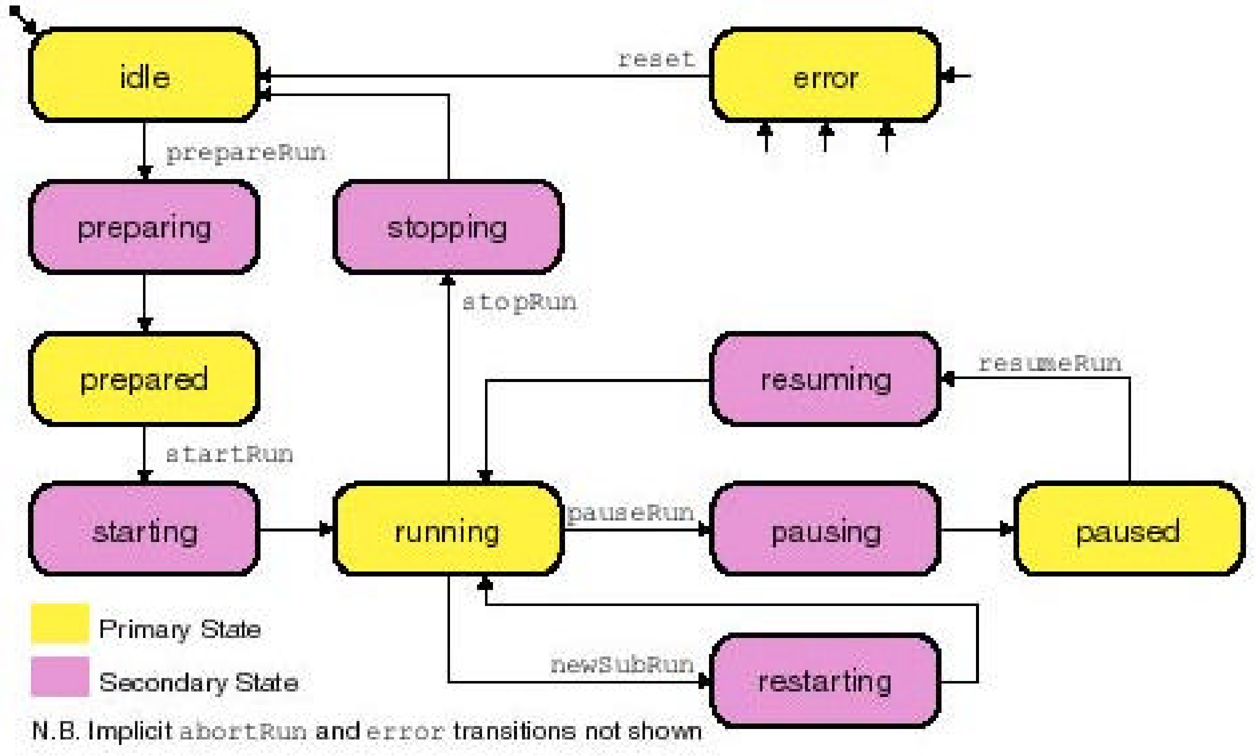
- Described as finite state machine:
  - Primary states: idle, running etc.
  - Secondary states: indicate transitions between primary (e.g. preparing, stopping)
  - Each component maintains own state
  - Run control has global state and issues transition commands (max one pending cmd)
- Unit of data-taking with “fixed” conditions is the run
- Sub-run for book-keeping purposes

Run Model States - Idle



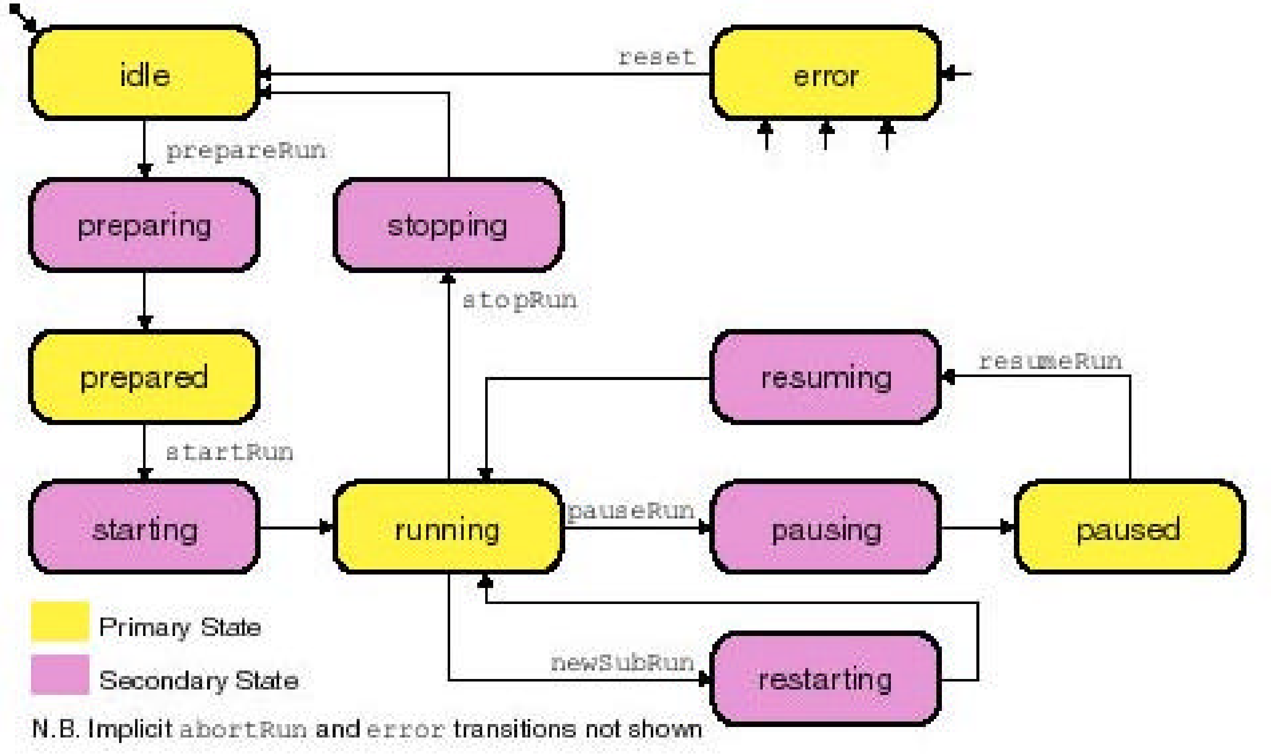
- default state, no run in progress or out of run

Run Model States - Prepared



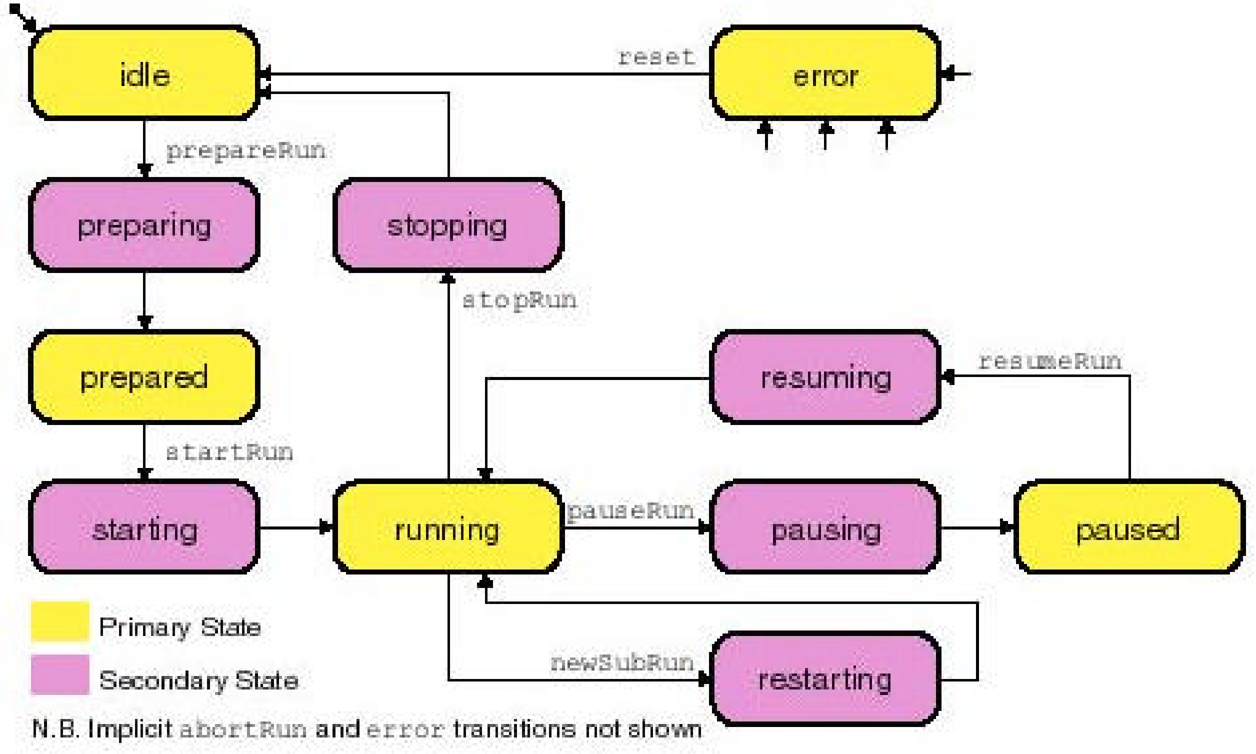
- component (& dependents) configured & prepared for data taking

Run Model States - Running



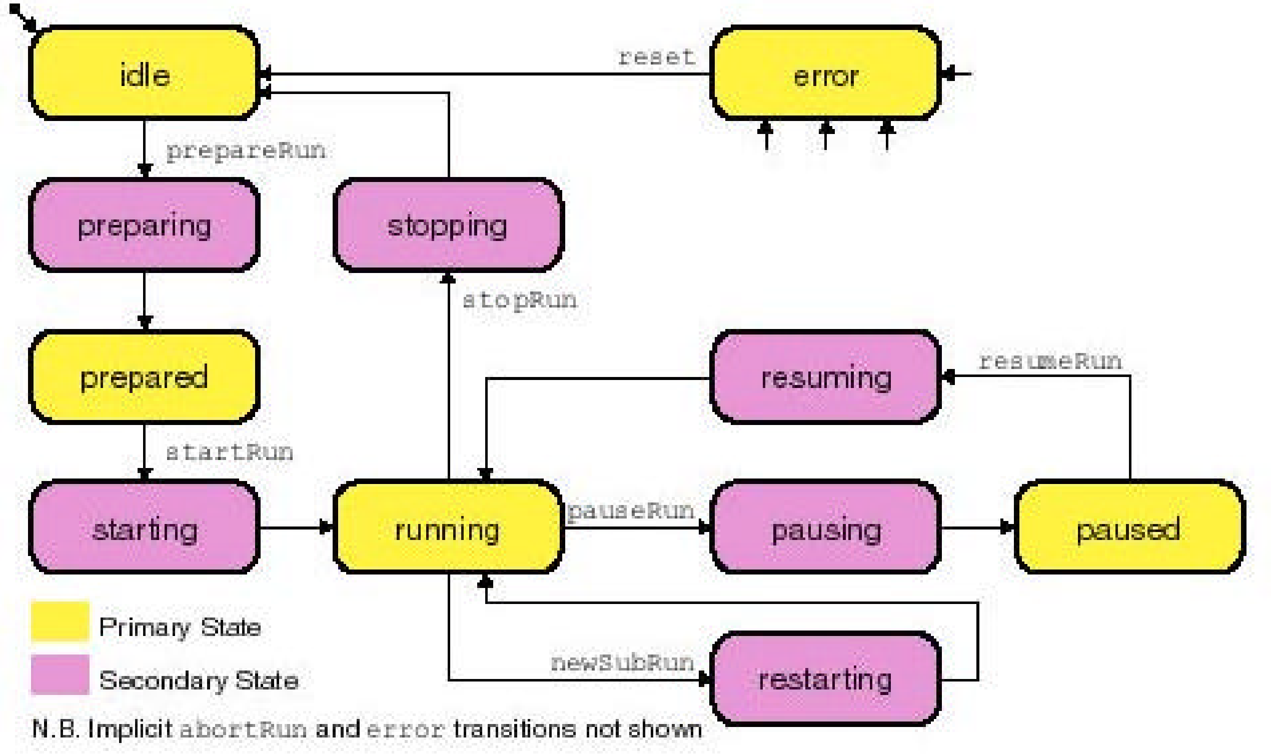
- data-taking in progress

Run Model States - Paused



- data-taking suspended (debugging mode)

Run Model States - Error



- one or more components failed to complete command etc.

Summary

- More details to follow:
  - Tass Belias: ROP, data transfer etc.
  - Mark Thomson: Run Control
- Still work to do resolving interfaces with other systems
- Hope to have clarified system & terminology!!